Comparison of Gaze-to-Objects Mapping Algorithms

Oleg Špakov
University of Tampere
Kanslerinrinne 1 B 1063
33014, Tampere, Finland
+358 3 3551 8556
oleg@cs.uta.fi

ABSTRACT

Gaze data processing is an important and necessary step in gaze-based applications. This study focuses on the comparison of several gaze-to-object mapping algorithms using various dwell times for selection and presenting targets of several types and sizes. Seven algorithms found in literature were compared against two newly designed algorithms. The study revealed that a fractional mapping algorithm (known) has produced the highest rate of correct selections and fastest selection times, but also the highest rate of incorrect selections. The dynamic competing algorithm (designed) has shown the next best result, but also high rate of incorrect selections. A small impact on the type of target to the calculated statistics has been observed. A strictly centered gazing has helped to increase the rate of correct selections for all algorithms and types of targets. The directions for further mapping algorithms improvement and future investigation have been explained.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Input devices and strategies

General Terms

Algorithms, Experimentation, Measurement, Performance, Design.

Keywords

Gaze to object mapping, eye gaze pointing and selection, algorithm design, gaze controlled applications.

1. INTRODUCTION

The development of gaze-controlled applications always deals with a search for gaze data correspondence to onscreen interactive objects. A solution of this task is not straightforward, as it is when the input is mouse cursor positions: even the most accurate eye-tracking systems designed for computer-aided assistance measure gaze point (a point on a screen where gaze lands on it) with some noise. Therefore, a user focus cannot be estimated using naïve mapping, when a gaze point reported by an eye-tracking system is always associated with the object that geometrically includes this point.

The noise presented in gaze point measurements during fixations consists of several components. The two obvious components are a) technology-related: the inaccuracy due to the imperfectness of "camera-image - to - screen" calibration-based transformation algorithms, and b) biology-related: noise due to various eye movements, such as tremor and microsaccades, and also a natural random offset (due to fuzzy fovea dimensions[1]) between the vector of actual attentive gaze direction and eye optical axe. Both components introduce dynamic noise into a measured gaze point; however, for a short period, like duration of a fixation, the noise can be roughly approximated as consisting of a constant (offset) and dynamic (jittering) components (for illustration, Figure 1 shows targets and recorded gaze points).

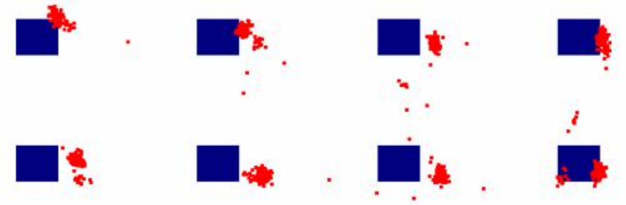

Figure 1. Noise in gaze data: offset and jittering.

Various methods have been proposed to overcome the inaccuracy problem in gaze point association with targets. It is worth to note that in many solutions fixations are supposed to be used as the input for mapping algorithms rather than raw gaze points. However, fixation detection algorithms are designed to classify all gaze points into "moving" (saccades) or "stable" (fixations), and answer a questions like "whether this gaze point is from the same fixation, as the previous one, or not?", rather than map them onto objects. Moreover, any fixation detection algorithm can be treated as a filter than outputs a mean of the collected gaze points, and therefore can be compared against gaze-to-object mapping algorithms. In this paper "gaze-point - to – object" mapping rather than "fixation-to-object" mapping is meant everywhere next.

The simplest and most evident solution to deal with the noise in gaze data is to make all interactive objects large enough, so that any inaccuracy in output from a gaze tracking system is compensated by object's size (for example, see the GazeTalk application developed by Hansen, Hansen and Johansen [2]). The only requirement for design of objects in this case is to make their visual informative part located at the very center, thus leaving a lot of empty (i.e., non-informative, but still included

NGCA'11, May 26–27, 2011, Karlskrona, Sweden.

[1] See more on this topic on http://webvision.med.utah.edu

into visual object representation) space between object's content and its borders.

A similar solution was proposed by Jacob [4]: the objects are large as well, but the visible object representation is limited to its informative part, thus leaving an invisible "gaze-sensible" area around (i.e., it exceeds the object's visual border). It gives the effect that objects "attract" gaze points. However, the screen space around objects cannot be filled by any information and should stay blank. Miniotas, Špakov and MacKenzie [8] studied dynamic expansion of gaze-sensible area: if a gaze point than belongs to a fixation lands on an object, the object is recognized to be in focus while the fixation continues (so called "grab-and-hold" effect). However, the requirement of having no objects around decreases the value of this solution. A similar to the "grab-and-hold" effect also was studied by Zhang, Ren, and Zha [16], who implemented object force-feedback and gaze point speed reduction algorithms to "hold" gaze over an object, thus preventing dwell time to reset.

Another mapping algorithm described by Monden, Matsumoto, Yamato [9] does not take into account the object's size, but rather associates a gaze point with the nearest object (the distance is calculated from object's center). However, in this case a number of unintended selections may be very high, as a user can make a selection looking at any point of a screen, even at its farthest from interactive objects place.

Graphical interfaces of applications are nearly always consists of interactive (buttons, menus, web-links, etc.) and passive (text, images, etc.) elements. The necessity to make all interactive elements large reduces the space left to accommodate non-interactive elements, and gaze-controlled interfaces sometimes contain specific interactive widgets to deal with this problem (for example, see [7] and [13]). It is also worth to note that non-interactive elements also can (and, probably, should) be designed so that their content does not take all the space available, but leave a gap between their informative content and border. Therefore, one may found that in many cases all screen real estate is divided between visual interface element (both interactive and passive), and algorithms that try to expand objects or split the "free" areas to assign to nearest elements will be out of use simply because there is no such "free" area, although they still may benefit in cases when a gaze point falls slightly out of a screen.

The only known option for this kind of gaze point mapping is a dynamic hidden resize of gaze-sensible area of interactive elements. The resize procedure uses some system or/and user states that it tracks. MacKenzie and Zhang [6] have introduced this kind of intelligence into their onscreen keyboard for eye-typing (operating with distances from fixation to object center, rather than with object sizes). In their keyboard the keys have various "priorities" at a certain time moment, so fixations are associating not with the nearest key, but rather with the key for which a product of distance and its priority gives the highest value. The follow-up study revealed that there was some (although very small) improvement in eye-typing (less errors, higher entry speed) using this mapping algorithm.

Another interesting proposal was introduced by Xu, Jiang, and Lau [15], who have suggested a complex association between a single gaze point and few nearest objects. In their study the authors were calculating an overall time spent focusing on objects, and if a gaze point was landed so that is was close to more than one object, its "time" (eye-tracking system sampling interval) was divided between all these objects proportionally to the distance to them. This idea may be found useful when applied to the gaze-based interaction using dwell time: the object that first accumulates a certain "amount of time", or "interest", will win the competition for the user's attention (focus) and get selected.

However, other known gaze point mapping algorithms used for dwell-time based selection always treat input data as solid temporal units. Instead, these algorithms put a specific attention of how to preserve the object's accumulated interest (i.e., sum of gaze points multiplied by sampling interval) if few gaze points have landed outside of the object. So, Hansen et al [3] has suggested simply to ignore such gaze points (i.e., do not alter the accumulated interest). Tien and Atkins [14] suggested to select objects if they collected 30 of 40 last gaze points, and then all of the next 20 gaze points (~700 ms). Almost the same proposal can be found in a study by Abe, Ohi, and Ohyama [1]: to be selected, an object has first ("initial" state) receive 5 consecutive gaze points (another options: more than half out of 5 recent gaze points), and then ("continuous" state) more than half out of 5 or 10 following points.

Laqua, Bandara, and Sasse [5] implemented a gaze-controlled application where objects were competing for the user's attention. Using the first solution, a gaze point contributed into some object's accumulated interest was not affecting the accumulated interest of other objects. The reset of accumulated interest values was initiated only at object (either any or the corresponding – two option were available) selection moment.

The other solution applies an accumulated interest decay effect to the objects other than the one onto which the recent gaze point was mapped. That is, each gaze point contributed into some object's accumulated interest was affecting the accumulated interest of other objects, in contrast to the previous solution. The accumulated interest was decreased by 50% for every 50 gaze points (1 second) mapped somewhere else. Decay of accumulated interest was also used by Ohno [10, 11] in his EyePrint application.

2. RESEARCH INTEREST

The variety of mapping algorithms which somehow deal with inaccuracy of output from eye-tracking systems naturally raises a question: which one is the best, if applied in real conditions, where objects are located next to each other and there is no space to expand their gaze-sensible area? Another research interest is related to the object size and its impact to the performance of these algorithms.

In existing gaze-controlled application objects usually presented as rectangles or circles with an appropriate pictogram. The impact of pictogram type (text, icon, or blank) to the performance of mapping algorithms is one more interest of this study.

It is expected, that user's focus, when looking at a target, is located at some distance from the target's center. The impact of this offset to the mapping performance is another target or this study.

The comparison required same gaze data to be streamed into mapping algorithms, therefore a database of gaze points was collected during an experiment, and then used to feed the comparing algorithms. Only the algorithms that can be used for gaze point mapping onto multiple adjacent objects and does not require any knowledge of the task, system or user states were selected for this study (the complete list is in the “Analysis” section).

3. DESIGNING A NEW ALGORITHM

The accumulated interest protection against an outlying gaze points was recognized as a key feature of all the algorithms described in the previous section. However, there is no agreement of the effect that an outlying gaze point should apply to the accumulated interest. In some cases, like force-feedback and speed-reduction mapping algorithm, the outlying gaze points reset the accumulated value. In another cases, like the one described in [1], the interest remains unchanged, even if it is evident that another object was focused. One more solution, like the second algorithm described in [5], implies slow decay of the interest.

The decision to decay somehow the accumulated interest seems to the most natural: the decay protects the accumulated interest against outlying gaze points, and resets it after some time, if another object got the user’s focus. However, the reported decay seems to be very slow, and more natural solution may sound as following: add the sampling interval to the interest of the object onto which a gaze point was mapped directly, and subtract same value from the interest of all other objects. The proposed algorithm was called a “competing algorithm”.

In case of often outlying gaze points, this kind of mapping algorithms will cause a delay in selection. Since outlying points can be recognized as “outlying” and not as “switch of focus” only after collecting several further points, it is possible to re-estimate their association with objects later. Thus, keeping a history of results of gaze point mappings and looking backward with every new gaze point may serve for improvement of the proposed algorithm. Naturally, the history should be limited in time and be reset after each selection.

The improved algorithm, named as “dynamic competing algorithm”, pulls in each past gaze point to the recent gaze point by the value inversely proportional to the distance and time interval between them. This procedure uses the Gaussian function and the parameters for the spatial and temporal components (sigma’s) was found empirically, but they may be dependent on peculiarities of output of the eye-tracking system used. The “pull-in” procedure resembles the magnetic effect of the algorithms used in [16].

4. METHOD

4.1 Experiment design

Three types of targets were used in this study: short words (3 letters), long words (9 letters), and icons. The monospaced font “Consolas” was used to display words, as it allows keeping their graphical representation of a same size. The lists of words were created from the “Oxford 3000” keywords[2] (the most frequent 3000 words of English language) in order to minimize cognitive load for participants, thus 3-letters list contained 177 words, and 9-letters list contained 253 words. The list of icons was constructed from various free icons (designed to the displayed on buttons) available in the Internet[3], and contained 195 icons.

A fourth type of object (“small dot”) was introduced to serve as a center of the targets emulated during the analysis step. It was done to make the comparison of centered and free gazing available: the size of emulated targets resembled the size of other shown targets.

The shown targets were appearing in three sizes: 18, 24, and 30 points for words, and 35, 60, and 85 pixels for square-shaped icons. The blank targets had corresponding size to each of graphical targets. A “small dot” target was displayed as a 6 by 6 pixels square. Words were displayed in black color, blank targets in grey color, and the screen background had dark grey color.

The screen was divided into 25 cells (5 by 5); targets were presenting in a center of each cell in random order. One block of the experiment consisted of presentation of 25 targets of the same type: only words (either from 3-letters or from 9-letters list), or icons, or blank targets of a same size were displayed within a block, but for each presentation a new random word or icon was used. The order of blocks was random.

4.2 Procedure

The first experiment step was a calibration of the eye tracker. Then a grid of 25 cells with circles (D=20px) at center was displayed, and a gaze point was visible on a screen. Participants where asked to quickly glance at each point. This procedure was introduced to estimate visually the quality of calibration. The displayed circles got a color between red and green, depending on what percentage of gaze points landed on a particular cell hit the central (invisible) square (D=50px). The calibration procedure was repeated if more than half of the circles were red or orange, until a satisfied calibration quality was recognized.

Then the first block started and a home box at the screen center was displayed for 2 seconds. The experiment continues displayed 25 targets one by one. Each target stood visible for 2 second; they were shown and hidden automatically. After a block was finished, the participants had a chance to rest, if needed, and then press a key to continue with the next block. This way all 20 blocks were completed: [3 types] x [3 sizes] x [graphical / blank] + 2 x [“small dot”]. Each participant gazed at 20 x 25 = 500 targets.

In order to escape visual search for small targets appearing far from the previously shown target, a big white rectangle was displayed for 150 ms at the target appearance location to make a “blink” effect.

The recorded data consisted of records, each containing a gaze points (timestamp and location) and the displayed target properties (location, size and type).

[2] http://www.oxfordadvancedlearnersdictionary.com/oxford3000

[3] In particular, some the icons located at http://wefunction.com/2008/07/function-free-icon-set/ were used in this study

4.3 Participants

Ten volunteers (aged 25–59 years, 5 male, 5 female) took part in the test. They were students or staff at the University of Tampere, and all had previously participated in other related eye typing experiments. Prior to the experiment, participants were informed about their rights and anonymity of the data in the experiment.

4.4 Equipment

The experiment was conducted in the usability laboratory at the University of Tampere. The Tobii T60 eye tracking system was used to measure participants' eye movements. The proprietary software with embedded ETU-Driver[4] was developed (C#, .NET 2.0, MS Visual Studio 2005) to display targets and collect data. The setup consisted adjustable chairs and tables. The chair was set so that the participant's eyes were at approximately 60 cm from the 17-inch monitor.

5. ANALYSIS

Altogether, 500 x 10 = 5000 sets of gaze points were recorded and used to feed the mapping algorithms.

The software dedicated for the first-step analysis was developed in C#, .NET 2.0, MS Visual Studio 2005[5]. It inputs the recorded data form a log file, and outputs results of selection for each displayed target. The results depend on the chosen dwell time, mapping algorithm, and chosen parameters of this algorithm. The output from this software consisted of 950 records when analyzing one log file (one participant data): 500 selections of the displayed targets, and 450 selections of emulated objects of all sizes when the target was presented as a "small dot". Each record contained location, size and type of target, movement time, selection time, and selection result (correct selection, incorrect selection, or no selection). This data was loaded into MS Excel to summarize the results of the study.

Four dwell times *DT* were used as selection criteria: 300, 500, 850 and 1300 milliseconds.

Most of the implemented algorithms use same additive-only interest calculation formula:

$$AT_t(O_i) = AT_{t-1}(O_i) + \begin{cases} S & if\,(\bar{x}_t, \bar{y}_t) \in O_i \\ 0 & otherwise \end{cases}$$

Here $AT_t(O_i)$ means the accumulated interest in milliseconds of the i^{th} object at the time t, S is the sampling interval in milliseconds, $(\bar{x}_t, \bar{y}_t)$ are the adjusted gaze point coordinates at the time t. The adjustment is algorithm-specific, and is shown below as the corresponding formula (if any) in the list of implemented algorithms (the interest calculation formula is shown also, if differs from the mentioned above):

- Competing, C

$$AT_t(O_i) = AT_{t-1}(O_i) + \begin{cases} S & if\,(\bar{x}_t, \bar{y}_t) \in O_i \\ -S & otherwise \end{cases},$$

where $\bar{x}_t = x_t$ and $\bar{y}_t = y_t$

- Dynamic competing, DC:

$$AT_t(O_i) = \sum_{j=0}^{t} \begin{cases} S & if\,(x_j, y_j) \in O_i \\ -S & otherwise \end{cases},$$

where $x_j = x_j + (x_t - x_j)e^{-\frac{(x_t-x_j)^2+(y_t-y_j)^2}{2s_s^2}}$,

$$y_j = y_j + (y_t - y_j)e^{-\frac{(x_t-x_j)^2+(y_t-y_j)^2}{2s_s^2}},$$

and $s_t = 80$, $s_s = 20$

- Force Feedback, FF [16]

$$\bar{x}_t = x_t + s\frac{D(\{x_t, y_t\}, \{\bar{x}_{t-1}, \bar{y}_{t-1}\})}{D(\{x_t, y_t\}, \{x_c, y_c\})}(x_t - x_c),$$

$$\bar{y}_t = y_t + s\frac{D(\{x_t, y_t\}, \{\bar{x}_{t-1}, \bar{y}_{t-1}\})}{D(\{x_t, y_t\}, \{x_c, y_c\})}(y_t - y_c),$$

where D is the distance between the given points, (x_c,y_c) is a center of object O_i, $s = 0.8$.

- Speed reduction, SR [16]

$$\begin{cases} \bar{x}_t = (1-r)x_t + rx_{t-1} \\ \bar{y}_t = (1-r)y_t + ry_{t-1} \end{cases} \quad if\ D(\{x_t, y_t\}, \{x_c, y_c\}) > D(\{x_{t-1}, y_{t-1}\}, \{x_c, y_c\}),$$

$$\begin{cases} \bar{x}_t = x_t \\ \bar{y}_t = y_t \end{cases} \quad otherwise$$

where D is the distance between the given points, (x_c,y_c) is a center of object O_i, and $r = 0.85$

- Fractional mapping, FM [15]

$$AT_t(O_i) = Se^{-\frac{(x_i-x_c)}{2s_x^2}-\frac{(y_i-y_c)}{2s_y^2}},$$

where (x_c,y_c) is a center of object O_i, $s_x = 120$, $s_y = 120$.

- Accurate ending, AE: 75% of 67% period of the dwell time, then 100% of the rest period [14].
- More-than-half, MTH: the "initial" state is 33% of the dwell time, and the "continuous" state is the rest period [1].
- Static interest accumulation, SIA [5].
- Dynamic interest decay, DID: 0.5% of accumulated interest for each 20 ms of gazing off-target [5]

$$AT_t(O_i) = AT_{t-1}(O_i) + \begin{cases} S & if\,(\bar{x}_t, \bar{y}_t) \in O_i \\ -D\frac{S}{20}AT_{t-1}(O_i) & otherwise \end{cases},$$

where $\bar{x}_t = x_t$, $\bar{y}_t = y_t$, and $D = 0.005$

To ensure the best performance of DC, FF, SR, FM and DID mapping algorithms, a prior search for their optimum parameters was completed (for illustration, the dependency of average rate of

[4] Available on http://www.cs.uta.fi/~oleg/

[5] Stimuli presentation and analysis software is available from http://www.cs.uta.fi/~oleg/products/LookPoint.zip

correct selection recorded using the FF algorithm on the parameter *s* is shown on Figure 2). The search was organized so that the lists of candidate values included those reported in the literature. The best parameter is the one that results in the highest rate of correct selection. However, if these rates were about equal for several parameters (the observed difference was less than 0.5%), then the selection time and rate of incorrectly selected objects were also taken into account to recognize the best parameter. The optimization was completed using *DT* = 500ms as a selection criterion.

No any data filtering was applied intentionally to prevent possible decrease in differences of the results between the tested algorithms and to determine they own efficiency when dealing with inaccuracies in gaze data.

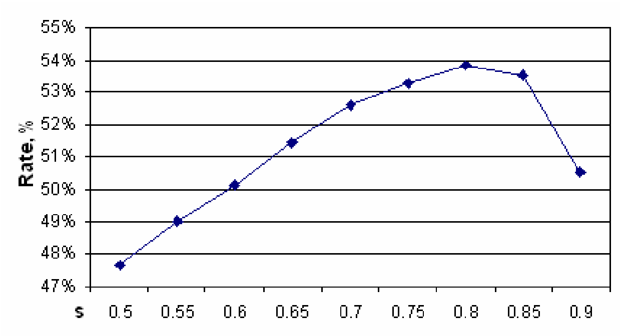

Figure 2. The rate of correct selections dependency on the parameter using FF algorithm.

The DC algorithm was the most demanding for computational power resulting in slow data processing, and the MTH algorithm was the fastest.

6. RESULTS AND DISCUSSION

The overall rate of correct selections was rather low (from 20% to 60%, about 45% in average) due to the small size of some targets and drifts in calibration. Therefore, a gaze data correction was introduced, and the mapping results of both original and corrected gaze data are presented next.

The correction was session and location -based: gaze data of each of 20 targets appeared at a particular location during same session was shifted equally. Applied offsets were calculated as a distance between the corresponding target center location and the average of (almost) all gaze points collect when a "small-dot" target was shown.

The grand mean of the rate of correct selections S_c is 46.8%, and the rate of incorrect selection S_w is 31.2%. The data correction has improved these values significantly: $\overline{S}_c$=65.9% and $\overline{S}_w$=15.6% ($p < 0.001$).

The performance of the algorithms is expressed as S_c, S_w, and selection time T_s (only for the cases when a selection was recognized): higher S_c, lower S_w, and shorter S_c point to better performance. The average S_c calculated for each algorithm using original data is shown in Figure 3 as colored solid bars. These value are quite similar if *DT*=300, but the Grubb's test treats the value of MTH's algorithm as an outlier, i.e. this algorithm performs worse than others at the given *DT*.

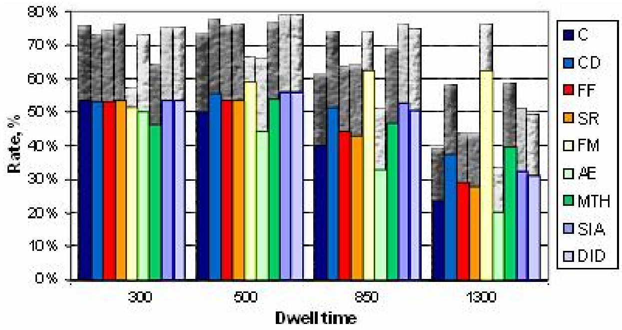

Figure 3. Correct selections, by algorithms and dwell time.

The increase of *DT* improves the S_c of FM algorithm, and worsens this rate of C and AE algorithms. Other algorithms reaches their highest S_c when *DT*=500, but then show its steady decrease as *DT* continues increasing and a time pressure (due to the approach to the duration of trials) become stronger. Some algorithms like C, SIA and DID suffer more from this increase than others like MTH.

The average S_w calculated for each algorithm using original data is shown in Figure 4 as colored solid bars. The C and AE are the most resistant algorithms to the selections of incorrect objects. The FM algorithm showed the worst resistance to the incorrect selections, but only when *DT* was 850 and 1300 ms. The DC and MTH algorithm also produces higher than average S_w for all *DT*s.

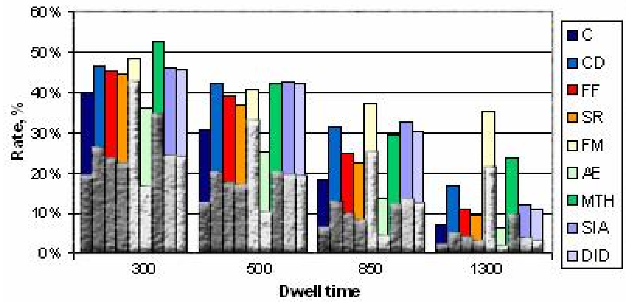

Figure 4. Incorrect selections, by algorithms and dwell time.

The DC and FM algorithms showed the shortest selection time independently from *DT*. Other algorithms had about equal T_s for same *DT*. These values are shown in Figure 5.

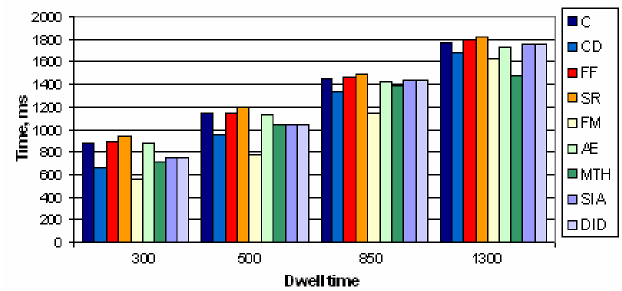

Figure 5. Selection time, by algorithms and dwell time.

The analysis of algorithms' performance reveals that the FM algorithm could be considered as the best because or the highest S_c and shortest T_s, and DC as the second best. However, both

Table 1. Rates of correct and incorrect selections and selection times by algorithms and target types using original (black) and corrected (blue) data

Algorithm			C		DC		FF		SR		FM		AE		MTH		SIA		DID	
S_c	Shown	Graphical	42.7	60.8	49.9	73.4	45.4	66.8	45.0	67.4	61.2	71.3	37.6	58.7	48.0	70.5	49.3	73.3	48.7	72.6
		Blank	41.1	59.8	49.1	68.3	44.8	62.2	44.2	62.8	56.6	65.8	36.8	53.7	45.4	63.7	48.2	67.8	47.6	67.0
		Mean	**41.9**	**60.3**	**49.5**	**70.9**	**45.1**	**64.5**	**44.6**	**65.1**	**58.9**	**68.5**	**37.2**	**56.2**	**46.7**	**67.1**	**48.8**	**70.5**	**48.1**	**69.8**
	Emulated		**44.3**	**78.8**	**52.7**	**87.0**	**48.4**	**77.6**	**48.0**	**81.6**	**60.6**	**73.1**	**39.1**	**71.8**	**46.5**	**76.9**	**51.4**	**85.4**	**50.6**	**84.7**
S_w	Shown	Graphical	23.9	8.5	34.4	14.4	30.7	12.6	28.8	11.5	38.3	27.9	19.6	6.7	37.1	17.0	33.7	13.6	32.6	13.2
		Blank	24.4	11.5	34.2	17.6	29.6	15.1	28.1	13.9	42.7	33.3	21.0	9.7	36.8	21.2	33.2	16.8	32.4	16.3
		Mean	**24.2**	**10.0**	**34.3**	**16.0**	**30.2**	**13.8**	**28.5**	**12.7**	**40.5**	**30.6**	**20.3**	**8.2**	**37.0**	**19.1**	**33.4**	**15.2**	**32.5**	**14.7**
	Emulated		**23.1**	**3.3**	**33.2**	**6.4**	**30.3**	**5.3**	**27.9**	**4.7**	**38.8**	**26.2**	**18.5**	**2.8**	**37.3**	**12.1**	**32.4**	**5.8**	**31.6**	**5.6**
S_w / S_c ratio			0.58	0.17	0.69	0.23	0.67	0.21	0.64	0.19	0.69	0.45	0.55	0.15	0.79	0.28	0.68	0.22	0.67	0.21
T_s	Shown	Graphical	1293	1267	1140	1125	1310	1300	1351	1338	1025	1023	1275	1254	1149	1137	1237	1210	1239	1213
		Blank	1332	1295	1170	1149	1340	1322	1375	1355	1034	1032	1315	1277	1165	1146	1266	1238	1267	1240
		Mean	**1312**	**1281**	**1155**	**1137**	**1325**	**1311**	**1363**	**1346**	**1030**	**1027**	**1295**	**1266**	**1157**	**1142**	**1252**	**1224**	**1253**	**1226**
	Emulated		**1313**	**1241**	**1149**	**1112**	**1329**	**1311**	**1368**	**1330**	**1037**	**1034**	**1291**	**1228**	**1149**	**1127**	**1253**	**1194**	**1255**	**1196**

algorithms showed high rate of S_w, and therefore their superiority is disputable. It is worth to note that the advantage in S_c at long DT resulted also in proportional disadvantage in S_w.

The ratio between S_w and S_c may help understanding the algorithms' resistance to the incorrect selection. This ratio is about equal for most of the algorithms, but is worse (greater) for MTH and better (lower) for C and AE algorithm. The latter algorithms showed the worst S_c, therefore they cannot be treated as candidates for the best performance.

The discussed ratios are shown in Table 1. The table also contains S_c, S_w and T_c values ranged by target type (rows) and algorithms (columns). Values calculated from original and corrected data are shown for each algorithm separately (left and right sub-columns, correspondingly).

The analysis of $\overline{S}_c$ and $\overline{S}_w$ using the corrected data reveals a bit distinctive picture. Although the FM algorithm still have much greater $\overline{S}_c$ when DT=1300, its $\overline{S}_c$ value for DT=300 is the worst (see Figure 3, grey rocky bars). Even more significant change was observed for its $\overline{S}_w$: the improvement in data quality resulted in very high values of $\overline{S}_w$ comparing to this value produced by other algorithms (see Figure 4, grey rocky bars), although in absolute units it decreased by 5-12%. Other algorithms showed about same increase of $\overline{S}_c$ and decrease of $\overline{S}_w$ comparing correspondingly to S_c and S_w. The ratio between $\overline{S}_w$ and $\overline{S}_c$ left lowest for AE and C algorithm, but the highest ratio got the FM algorithm: it was about twice greater than the average.

The analysis of algorithms' S_c dependency on target size reveals that the observed differences in this value between algorithms are strongly expressed if target size is small, and this difference vanishes for the largest targets, as expected. For example, the FM algorithm showed about twice higher S_c than others algorithm when targets were words displayed in small font. The rates of correct selection for all targets and algorithms

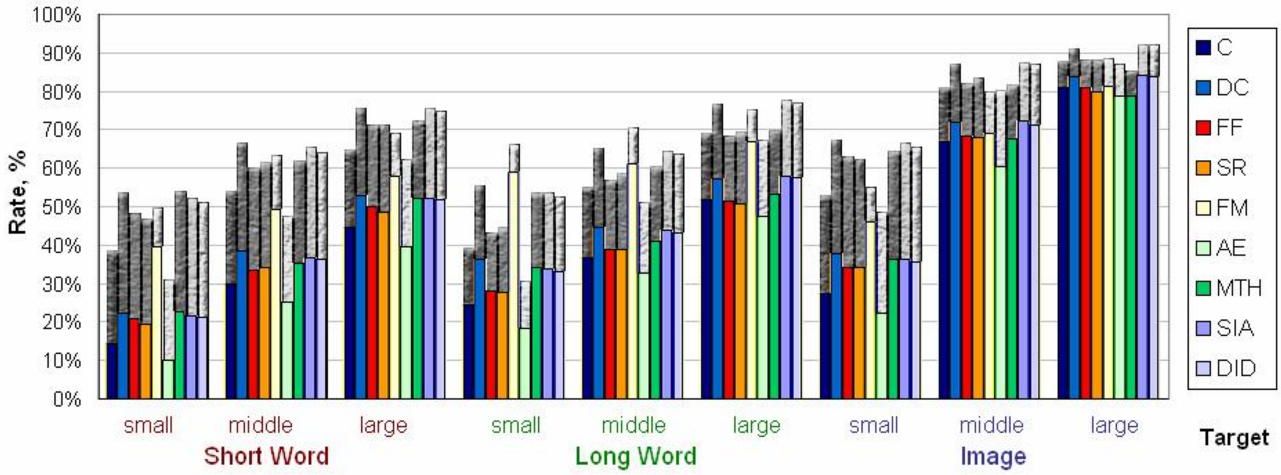

Figure 6. Correct selections, by algorithms and targets.

are shown in Figure 6: the colored solid bars denote the results of analysis using original data, and grey rocky bars denote the results of analysis using corrected data. However, the discrimination in S_w is also dependent on the target size in a similar way. Similar dependency was observed when analyzing both selection rates using corrected data.

The summarized impact of target's pictogram is shown in Figure 7. The S_c rates are slightly lower for the blank targets: 46.0% versus 47.5% in average for the graphical targets using the original data and 63.5% versus 68.5% in average using the corrected data, and the differences are statistically significant ($p < 0.01$ and $p < 0.001$ correspondingly). The type of target affected the S_c and S_w values of each algorithm about equally.

The direct comparison between the results produced for the trials with long and short words is not applicable, as they occupied distinct amount of screen. However, the visual inspection of the values shown in Figure 6 does not reveal any remarkable changes of the relative performance of the tested algorithms.

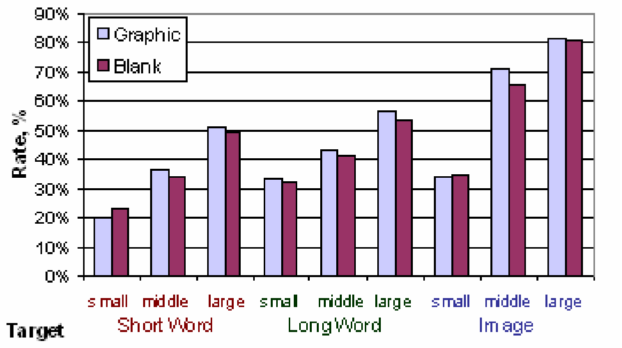

Figure 7. Correct selections, by targets.

The grand mean of S_c for emulated targets was slightly higher than for shown targets (47.9% versus 45.6%, $p < 0.001$) when making analysis with the original data. This improvement was about equal for many algorithms (but for MTH algorithm there was no improvement at all, $p < 0.05$) and types of targets, therefore the centralized gaze allocation over a target was recognized as helping in selecting it, but the benefit was only about 5%. However, the increase of S_c for large images comparing to S_c of small and middle-sized images was much higher (5.7% against 0.8% and 0.3%) and statistically significant ($p < 0.001$). The effect of gaze centralized position was the opposite when using the corrected data: the increase of S_c for large images was only 3.8% against 19.4% and 7.2%; for other types of targets the improvement consisted of about 15%.

The improvement of S_c is higher if the corrected data was used, but this "artificial" increase was expected, as same data used for emulation and correction. Nevertheless, the differences in changes of S_c still offer the interest. Indeed, the improvement of S_c differed a lot among the algorithms. So, the average improvement was 13.8%, but the FM algorithm improved its S_c only by 4.6%, while the C algorithm improved S_c by 18.5% (see Figure 8). A weak decrease of S_w (about one percent) in the analysis with original data also has changed a lot (7.6% in average) when the corrected data was used.

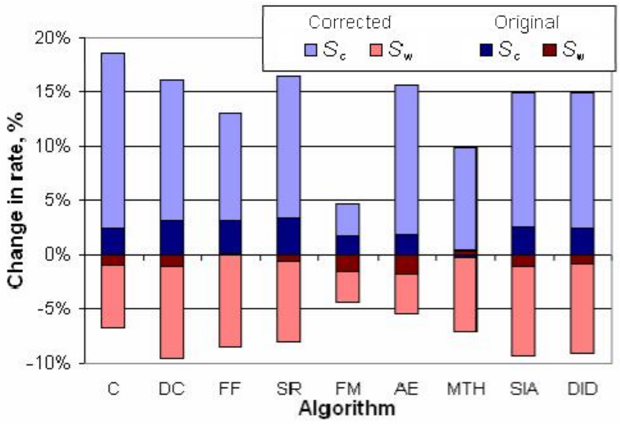

Figure 8. Changes in rates of correct and incorrect selections, by algorithm.

7. CONCLUSIONS

The fractional mapping algorithm FM was recognized as the best one if considering the rate of correction selection S_c (especially when selecting small targets and the time pressure was high) and selection time T_s: for this algorithm these values were the best in most of the tested conditions. However, its rate of correction selection S_w was often the highest. The ratio between S_w and S_c was similar to those of other algorithm, therefore the functionality of this algorithm can be treated as leading to simply a highest rate of selections: this algorithm allowed to make a selection even when all other algorithms gave up to do it within the given time window. This observation does not allow treating this algorithm as the ultimate winner of the conducted competition.

The designed dynamic competing algorithm DC has shown a good performance: its S_c and T_s were one of the best in most of the conditions, and less sensitive to the time pressure at long dwell times than for other algorithms. However, it also showed high rate of incorrect selections.

The S_c averaged by all algorithms was slightly lower for the blank targets, and slightly higher for the targets with central gaze allocation. Only few algorithms had distinct effect on the type of targets than the majority. However, the study leads to a conclusion that the design of target graphical view so that it attracts gaze to its very central location is advisable.

8. FUTURE WORK

The current study was conducted using an expensive high-end eye tracking system. The low-cost eye tracking systems with free software and off-the-shelf hardware components available nowadays are expected producing data of less quality, and therefore the performance of the tested algorithms may differ significantly. The future work includes testing the algorithms on data gathered from systems like ITU GazeTracker [12].

Another direction of this research is the investigation of possibilities for further improvement of some algorithms. For example, the combination of DC, FM and some other algorithms may lead to a very high S_c rates with low T_s and moderate S_w.

Finally, the impact of a prior data filtering and smoothing is one more interest of the research in gaze-to-object mapping.

9. ACKNOWLEDGMENTS

This work was supported by Academy of Finland (grant 256415).

10. REFERENCES

[1] Abe, K., Ohi, S., and Ohyama, M. 2007. An Eye-Gaze Input System Using Information on Eye Movement History. In *Proceedings of the 4th international conference on Universal access in human-computer interaction: ambient interaction (UAHCI '07)*, Stephanidis, C. (Ed.). Springer-Verlag, Berlin, Heidelberg, 721-729.

[2] Hansen, J. P., Hansen, D. W., and Johansen, A. S. 2001. Bringing Gaze-based Interaction Back to Basics. In *Stephanidis, C. (Ed.), Universal Access in Human-Computer Interaction*, Lawrence Erlbaum Associates, 325-328.

[3] Hansen, J.P., Johansen, A.S., Hansen, D.W., Itoh, K., and Mashino, S. 2003. Command without a click: dwell time typing by mouse and gaze selections. In *Proceedings of Human Computer Interaction (INTERACT '03)*, IOS Press, Amsterdam, 121-128.

[4] Jacob, R. J. K. (1995) Eye tracking in advanced interface design. In *Virtual environments and advanced interface design, Woodrow Barfield and Thomas A. Furness, III (Eds.).* Oxford University Press, Inc., New York, NY, USA, 258-288.

[5] Laqua, S., Bandara, S. U. and Sasse, M. A. 2007. GazeSpace: eye gaze controlled content spaces. In *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...but not as we know it (BCS-HCI '07)*, Vol. 2. British Computer Society, Swinton, UK, 55-58.

[6] MacKenzie, I. S., and Zhang, X. (2008) Eye typing using word and letter prediction and a fixation algorithm. In *Proceedings of the 2008 symposium on Eye tracking research & applications (ETRA '08).* ACM, New York, NY, USA, 55-58.

[7] Miniotas, D., Špakov, O., and Evreinov, G. 2003. Symbol Creator: An Alternative Eye-Based Text Entry Technique with Low Demand for Screen Space. In *Proceedings of IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2003)*, IOS Press, pp. 137-143.

[8] Miniotas, D., Špakov, O., and MacKenzie. I.S. 2004. Eye gaze interaction with expanding targets. In *Extended abstracts on Human factors in computing systems (CHI '04).* ACM, New York, NY, USA, 1255-1258.

[9] Monden, A., Matsumoto, K., and Yamato, M. 2005. Evaluation of gaze-added target selection methods suitable for general GUIs. In *International Journal on Computer Applications in Technology*, 24(1), 17-24.

[10] Ohno, T. 2001. EyePrint - An Information Navigation Method with Gaze Position Trace. In *Human-Computer Interaction (INTERACT '01)*, Hirose M. (ed.), IOS Press, 743-744.

[11] Ohno, T. (2004) EyePrint: Support of Document Browsing with Eye Gaze Trace, In *Proceedings of the 6th International Conference on Multimodal Interfaces (ICMI'04)*, 16-23.

[12] San Agustin, J., Skovsgaard, H., Hansen, J. P., and Hansen, D. W. 2009. Low-cost gaze interaction: ready to deliver the promises. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems (CHI EA '09).* ACM, New York, NY, USA, 4453-4458.

[13] Špakov, O., and Majaranta, P. 2009. Scrollable Keyboard for Casual Gaze Typing. In *Boehme, M., Hansen, J.-P., and Mulvey, F. (eds.) PsychNology Journal: Gaze Control for Work and Play*, ISSN 1720-7525, 7(2), pp. 159-173.

[14] Tien, G., and Atkins. M. S. 2008. Improving hands-free menu selection using eyegaze glances and fixations. In *Proceedings of the 2008 symposium on Eye tracking research & applications (ETRA '08).* ACM, New York, NY, USA, 47-50.

[15] Xu, S., Jiang, H., and Lau, F. C. M. 2008. Personalized online document, image and video recommendation via commodity eye-tracking. In *Proceedings of the 2008 ACM conference on Recommender systems (RecSys '08)*. ACM, New York, NY, USA, 83-90.

[16] Zhang, X., Ren, X., and Zha. H. 2008. Improving eye cursor's stability for eye pointing tasks. In *Proceeding of the 26th annual SIGCHI conference on Human factors in computing systems*, ACM, Florence, Italy, 525–534.